Nonlinear Dyn (2017) 87:2235–2249
DOI 10.1007/s11071-016-3186-2

ORIGINAL PAPER

Switching processes in polynomiography

Krzysztof Gdawiec

Received: 24 May 2016 / Accepted: 1 November 2016 / Published online: 11 November 2016

Abstract Mandelbrot and Julia sets are examples of fractal patterns generated in the complex plane. In the literature, we can find many generalizations of those sets. One of such generalizations is the use of switching process. In this paper, we introduce some switching processes to another type of complex fractals, namely polynomiographs. Polynomiograph is an image presenting the visualization of the complex polynomial's root finding process. The proposed switching processes will be divided into four groups, i.e., switching of: the root finding methods, the iterations, the polynomials and the convergence tests. All the proposed switching processes change the dynamics of the root finding process and allow us to obtain new and diverse fractal patterns.

Keywords Dynamics · Polynomiography · Switching process · Fractal

1 Introduction

A very common method of generating fractal patterns in the complex plane is repeated iteration of a complex function $f : \mathbb{C} \to \mathbb{C}$ in the following way:

$$z_{n+1} = f(z_n), \tag{1}$$

K. Gdawiec (✉)
Institute of Computer Science, University of Silesia,
Będzińska 39, 41-200 Sosnowiec, Poland
e-mail: kgdawiec@ux2.math.us.edu.pl

where f depends on some constant $c \in \mathbb{C}$ and $z_0 \in \mathbb{C}$ is the point from a considered area of the complex plane [16]. Two examples of such fractals are Mandelbrot and Julia sets together with their different generalizations.

One of the generalizations is the use of switching processes. The study of this type of processes in Mandelbrot and Julia sets began in (1991) in work of Lakhtakia [13]. He introduced the following switching process:

$$z_{n+1} = \begin{cases} z_n^p + c_1, & \text{if} \quad |z_n| \leq r_0, \\ z_n^q + c_2, & \text{if} \quad |z_n| > r_0, \end{cases} \tag{2}$$

where p, q are natural numbers greater than 1, $c_1, c_2 \in \mathbb{C}$ and $r_0 > 0$ acts as a buffer between the two processes.

From the introduction of the switching process by Lakhtakia in the literature appeared several different switching process:

– Shirriff in (1993) introduced the following process [16]:

$$z_{n+1} = \begin{cases} z_n^{k_1} + h_1(c), & \text{if} \quad n \text{ is odd}, \\ z_n^{k_2} + h_2(c), & \text{if} \quad n \text{ is even}, \end{cases} \tag{3}$$

where $k_1, k_2 \in \mathbb{N}$, $h_1, h_2 : \mathbb{C} \to \mathbb{C}$ are mappings and $c \in \mathbb{C}$,
– Xing-Yuan in (2003) introduced the following process [19]:

$$z_{n+1} = \begin{cases} z_n^p + c, & \text{if} \quad |z_n| \le r_0, \\ z_n^q + c, & \text{if} \quad |z_n| > r_0, \end{cases} \tag{4}$$

where $p, q \in \mathbb{C}$, $\Re(p), \Re(q) \notin [0, 1]$ ($\Re(z)$ is the real part of z), $c \in \mathbb{C}$ and $r_0 > 0$,

– Xingyuan in (2006) introduced the following process [20]:

$$z_{n+1} = \begin{cases} z_n^{\alpha_0} + h_0(c), & \text{if} \quad n+1 \mod m = 0, \\ z_n^{\alpha_1} + h_1(c), & \text{if} \quad n+1 \mod m = 1, \\ \dots \\ z_n^{\alpha_{m-1}} + h_{m-1}(c), & \text{if} \quad n+1 \mod m = m-1, \end{cases} \tag{5}$$

where $m \ge 2$, $\alpha_j \in \mathbb{R}$ and $h_j : \mathbb{C} \to \mathbb{C}$ are mappings for $j = 0, 1, \dots, m-1$, and $c \in \mathbb{C}$, $z_0 = h_0(c)$,

– Ashlock and Jamieson in (2008) introduced the following process [2]:

$$z_{n+1} = \begin{cases} z_n^2 + c_1, & \text{if} \quad n+1 \mod 2 = 1, \\ z_n^2 + c_2, & \text{if} \quad n+1 \mod 2 = 0, \end{cases} \tag{6}$$

where $c_1, c_2 \in \mathbb{C}$. This process was later studied, e.g., in [3,4,18],

– Negi et al. in (2008) introduced the following process [14]:

$$z_{n+1} = \begin{cases} s(z_n^p + c_1) + (1-s)z_n, & \text{if} \quad |z_n| \le r_0, \\ s(z_n^q + c_2) + (1-s)z_n, & \text{if} \quad |z_n| > r_0, \end{cases} \tag{7}$$

where $s \in [0, 1]$, $p, q \in \mathbb{N}$, $c_1, c_2 \in \mathbb{C}$ and $r_0 > 0$,

– Yadav and Rani in (2015) introduced the following process [21]:

$$z_{n+1} = \begin{cases} s(z_n^p + c_1) + (1-s)z_n, & \text{if} \quad n \text{ even}, \\ s(z_n^p + c_2) + (1-s)z_n, & \text{if} \quad n \text{ odd}, \end{cases} \tag{8}$$

where $s \in [0, 1]$, $p \in \{2, 3\}$ and $c_1, c_2 \in \mathbb{C}$.

Another example of fractal pattern that is generated in the complex plane is polynomiograph [11]. This method is based on iteration of a given root finding method. In this paper, we introduce into polynomiography several switching processes and show some examples of fractal pattern obtained with their help.

The remainder of this paper is outlined as follows. In Sect. 2, some basic information about polynomiography is presented, starting from the root finding methods, going through the different iteration process and convergence tests and ending up with the basic polynomiograph's generation algorithm. Sect. 3 is devoted to the introduction into polynomiography of the different switching processes. Next, in Sect. 4 some polynomiographs obtained with the help of the proposed switching processes and their generation times are presented. Finally, in Sect. 5, we give some concluding remarks.

2 Polynomiography

Polynomiography is defined as the art and science of visualization in approximation of the zeros of complex polynomials, via fractal and non-fractal images created using the mathematical convergence properties of iteration functions [11]. A single image created using the mentioned methods is called a polynomiograph.

The main element of polynomiograph's generation algorithm is the root finding method. In the literature, we can find many different methods of finding the roots of a polynomial, e.g., Newton's method [10], harmonic mean Newton's method [1], Whittaker's method [17], Traub–Ostrowski method [17]. Let us recall some of the methods that later will be used in the examples.

Let $p \in \mathbb{C}[Z]$, $\deg p \ge 2$ be a polynomial of the form:

$$p(z) = a_n z^n + a_{n-1} z^{n-1} + \dots + a_1 z + a_0. \tag{9}$$

Define a sequence of functions $D_m : \mathbb{C} \to \mathbb{C}$ in the following way [12]: $D_0(z) = 1$ and for $m > 0$ let

$$D_m(z) = \det \begin{bmatrix} p'(z) & \frac{p''(z)}{2!} & \dots & \frac{p^{(m-1)}(z)}{(m-1)!} & \frac{p^{(m)}(z)}{m!} \\ p(z) & p'(z) & \ddots & \ddots & \frac{p^{(m-1)}(z)}{(m-1)!} \\ 0 & p(z) & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \frac{p''(z)}{2!} \\ 0 & 0 & \dots & p(z) & p'(z) \end{bmatrix}. \tag{10}$$

Using the D_m sequence, we define a family of root finding methods, the so-called parametric basic family of iterations [12], in the following way:

$$B_{m,\lambda}(z) = z - \lambda p(z)\frac{D_{m-2}(z)}{D_{m-1}(z)}, \tag{11}$$

where $m = 2, 3, \ldots$ and $\lambda \in \mathbb{C}$.

Let us see how the first three elements of the parametric basic family of iterations look like:

$$B_{2,\lambda}(z) = z - \lambda\frac{p(z)}{p'(z)}, \tag{12}$$

$$B_{3,\lambda}(z) = z - \lambda\frac{2p'(z)p(z)}{2p'(z)^2 - p''(z)p(z)}, \tag{13}$$

$$B_{4,\lambda}(z) = z - \lambda\frac{6p'(z)^2p(z) - 3p''(z)p(z)^2}{p'''(z)p(z)^2 + 6p'(z)^3 - 6p''(z)p'(z)p(z)}. \tag{14}$$

One can easily see that $B_{2,\lambda}$ is the relaxed Newton's method [10], whereas $B_{3,\lambda}$ is the relaxed Halley's method [15].

Now, let us draw our attention to the Euler–Schröder family of iterations [12]. The initial elements of this family have the following form:

$$E_2(z) = z - \frac{p(z)}{p'(z)}, \tag{15}$$

$$E_3(z) = E_2(z) + \left(\frac{p(z)}{p'(z)}\right)^2\frac{p''(z)}{2p'(z)}, \tag{16}$$

$$E_4(z) = E_3(z) - \left(\frac{p(z)}{p'(z)}\right)^3\left(\frac{p'''(z)}{6p'(z)} - \frac{p''(z)}{2p'^2(z)}\right), \tag{17}$$

$$E_5(z) = E_4(z) + \left(\frac{p(z)}{p'(z)}\right)^4\left(\frac{p^{IV}(z)}{4!p'(z)} - \frac{5p''(z)p'''(z)}{12p'^2(z)} + \frac{5p''^3(z)}{8p'^3(z)}\right). \tag{18}$$

One can easily see that E_2 is the Newton's method. The detailed construction of the other elements of the family can be found in [12].

In the standard polynomiograph's generation algorithm, proposed by Kalantari, we use the feedback iteration (1), where as the function f we take a given root finding method. This type of iteration is often called the Picard iteration. We can extend the algorithm to other iteration processes known from fixed point theory [9]. Examples of such iterations are:

– Mann iteration

$$z_{n+1} = (1-\alpha_n)z_n + \alpha_n f(z_n), \quad n = 0, 1, 2, \ldots, \tag{19}$$

where $\alpha_n \in [0, 1]$ for all $n \in \mathbb{N}$,

– Noor iteration

$$\begin{cases} z_{n+1} = (1-\alpha_n)z_n + \alpha_n f(u_n), \\ u_n = (1-\beta_n)z_n + \beta_n f(v_n), \\ v_n = (1-\gamma_n)z_n + \gamma_n f(z_n), \quad n = 0, 1, 2, \ldots, \end{cases} \tag{20}$$

where $\alpha_n, \beta_n, \gamma_n \in [0, 1]$ for all $n \in \mathbb{N}$,

– CR iteration

$$\begin{cases} z_{n+1} = (1-\alpha_n)u_n + \alpha_n f(u_n), \\ u_n = (1-\beta_n)f(z_n) + \beta_n f(v_n), \\ v_n = (1-\gamma_n)z_n + \gamma_n f(z_n), \quad n = 0, 1, 2, \ldots, \end{cases} \tag{21}$$

where $\alpha_n, \beta_n, \gamma_n \in [0, 1]$ for all $n \in \mathbb{N}$ and $\sum_{n=0}^{\infty}\alpha_n = \infty$,

– Karakaya iteration

$$\begin{cases} z_{n+1} = (1-\alpha_n-\beta_n)u_n + \alpha_n f(u_n) + \beta_n f(v_n), \\ u_n = (1-a_n-b_n)v_n + a_n f(v_n) + b_n f(z_n), \\ v_n = (1-\gamma_n)z_n + \gamma_n f(z_n), \quad n = 0, 1, 2, \ldots, \end{cases} \tag{22}$$

where $\alpha_n, \beta_n, \gamma_n, a_n, b_n \in [0, 1]$, $\alpha_n + \beta_n \in [0, 1]$, $a_n + b_n \in [0, 1]$ for all $n \in \mathbb{N}$ and $\sum_{n=0}^{\infty}(\alpha_n + \beta_n) = \infty$.

Let us notice that for certain values of the parameters some of the iterations reduce to the others, e.g., Noor iteration reduces to Mann iteration for $\beta_n = \gamma_n = 0$.

When generating Mandelbrot or Julia sets, we stop the iteration process when the escape criterion is meet. In polynomiography, we stop iterating when we have found the root. The two standard tests (convergence tests) that are usually used have the following form:

$$|z_{n+1} - z_n| < \varepsilon, \tag{23}$$

$$|p(z_{n+1})| < \varepsilon, \tag{24}$$

where $\varepsilon > 0$ is the accuracy of computations. In [7] convergence tests that are based on metric and non-metric conditions were introduced, and they were used to obtain new artistic fractal patterns. Examples of such tests are the following:

$$|0.01(z_{n+1} - z_n)| + |0.029|z_{n+1}|^2 - 0.03|z_n|^2| < \varepsilon, \tag{25}$$

$$\left| \frac{0.045}{|z_{n+1}|^2} - \frac{0.05}{|z_n|^2} \right| < \varepsilon. \tag{26}$$

The pseudocode of polynomiograph's generation method is presented in Algorithm 1. Iteration I_q in the algorithm is one of the iteration methods from fixed point theory, and q is the vector of its parameters. The iteration takes three arguments: the root finding method R, the polynomial p for which we are looking the roots and the point z_n from the previous iteration. Convergence test T_t is one of the standard convergence tests or the ones from [7], and t is the vector of its parameters, e.g., ε, constants in (25) or (26).

Algorithm 1: Polynomiograph generation

Input: $p \in \mathbb{C}[Z]$ – polynomial, $A \subset \mathbb{C}$ – area, K – maximum number of iterations, I_q – iteration method, $q \in \mathbb{C}^N$ – parameters of the iteration I_q, R – root finding method, T_t – convergence test, $t \in \mathbb{R}^M$ – parameters of the convergence test T_t.
Output: Polynomiograph for the area A.

```
for z_0 ∈ A do
    n = 0
    while n ≤ K do
        z_{n+1} = I_q(R, p, z_n)
        if T_t(z_n, z_{n+1}) = true then
            break
        n = n + 1
    determine the color for z_0
```

To generate polynomiographs, we can also use other generation algorithms. For instance, in [8] we can find methods that are based on the well-known generation algorithms of Mandelbrot and Julia sets.

3 Switching processes in polynomiography

In the introduction, we have seen several switching processes that were used in the generation of Mandelbrot and Julia sets. In polynomiography, we can also introduce similar switching processes and propose new ones. We can divide the switching processes into four groups:

1. switching of the root finding methods,
2. switching of the iterations,
3. switching of the polynomials,
4. switching of the convergence tests.

In the first group in the polynomiograph's generation algorithm, we fix all the parameters except the root finding method. Depending on the switching method, we choose two or more root finding methods $R_0, R_1, \ldots, R_{m-1}$ ($m \geq 2$). We define the switching processes of the root finding methods as follows:

– switching modulo m

$$z_{n+1} = \begin{cases} I_q(R_0, p, z_n), & \text{if } n \mod m = 0, \\ I_q(R_1, p, z_n), & \text{if } n \mod m = 1, \\ \ldots & \\ I_q(R_{m-1}, p, z_n), & \text{if } n \mod m = m-1, \end{cases} \tag{27}$$

– switching using $|z|$

$$z_{n+1} = \begin{cases} I_q(R_0, p, z_n), & \text{if } |z_n| \leq r, \\ I_q(R_1, p, z_n), & \text{if } |z_n| > r, \end{cases} \tag{28}$$

where $r > 0$,

– switching using $|p(z)|$

$$z_{n+1} = \begin{cases} I_q(R_0, p, z_n), & \text{if } |p(z_n)| \leq r, \\ I_q(R_1, p, z_n), & \text{if } |p(z_n)| > r, \end{cases} \tag{29}$$

where $r > 0$.

The first two types of switching processes are very similar to the switching processes used in the generation of Mandelbrot and Julia sets. In the third process, we switch between two root finding methods depending on the distance from a root. If the distance of polynomial's value in the approximation point (from the previous iteration) to 0 is smaller than r, then we use one root finding method and the second one otherwise.

In the second group of switching processes in the polynomiograph's generation algorithm, we fix all the parameters except the iteration process. We choose two or more iteration processes $I^0_{q_0}, I^1_{q_1}, \ldots, I^{m-1}_{q_{m-1}}$ ($m \geq$

2), but the choice cannot be completely arbitrary. In Sect. 2, we noticed that for certain values of the parameters some of the iterations reduce to the others. If we choose two such iteration, then there will be no switching process. Thus, we need to choose iterations processes that do not reduce to each other for any possible combination of the parameters. In Fig. 1, we see diagram of dependencies between iterations presented in [9]. Two iterations from this diagram reduce to each other if there is a path in the diagram from one iteration to the other. So, two iterations do not reduce if there is no path in the diagram between them. When we have chosen the iteration processes, then we can define the switching processes between them as follows:

– switching modulo m

$$z_{n+1} = \begin{cases} I^0_{q_0}(R, p, z_n), & \text{if } n \mod m = 0, \\ I^1_{q_1}(R, p, z_n), & \text{if } n \mod m = 1, \\ \dots & \\ I^{m-1}_{q_{m-1}}(R, p, z_n), & \text{if } n \mod m = m-1, \end{cases} \tag{30}$$

– switching using $|z|$

$$z_{n+1} = \begin{cases} I^0_{q_0}(R, p, z_n), & \text{if } |z_n| \le r, \\ I^1_{q_1}(R, p, z_n), & \text{if } |z_n| > r, \end{cases} \tag{31}$$

where $r > 0$,

– switching using $|p(z)|$

$$z_{n+1} = \begin{cases} I^0_{q_0}(R, p, z_n), & \text{if } |p(z_n)| \le r, \\ I^1_{q_1}(R, p, z_n), & \text{if } |p(z_n)| > r, \end{cases} \tag{32}$$

where $r > 0$.

In the third group of switching processes in the polynomiograph's generation algorithm, we also fix all the parameters except one. This time we will switch between two polynomials p_1, p_2. Similarly like in the case of iterations, we cannot use arbitrary polynomials for the switching process. If we use polynomials for which the roots of one polynomial are distant from the roots of the other, then the root finding method in one iteration will try to converge to roots of one of the polynomials and in the other iteration to the roots of the other polynomial and in consequence it might happen that the method will not converge to any root.

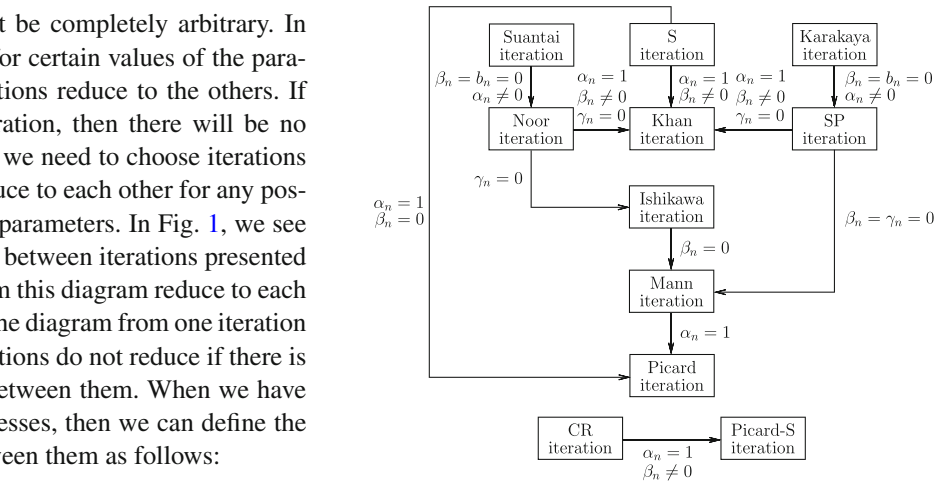

Fig. 1 Diagram of the iterations' dependencies

Therefore, we propose following way of choosing the polynomials. The first polynomial p_1 is chosen arbitrarily. Let $S_1 = \{r_1, r_2, \dots, r_k\}$ be the set of roots of p_1. From S_1 we select a subset $S_2 = \{r^*_1, r^*_2, \dots, r^*_l\}$, where $l < k$ and $r^*_i \in S_1$ for $i = 1, 2, \dots, l$. Then, polynomial p_2 is defined as follows:

$$p_2(z) = (z - r^*_1)(z - r^*_2) \dots (z - r^*_l). \tag{33}$$

Having polynomials p_1 and p_2, we define the switching processes in the following way:

– switching modulo 2

$$z_{n+1} = \begin{cases} I_q(R, p_1, z_n), & \text{if } n \text{ is even}, \\ I_q(R, p_2, z_n), & \text{if } n \text{ is odd}, \end{cases} \tag{34}$$

– switching using $|z|$

$$z_{n+1} = \begin{cases} I_q(R, p_1, z_n), & \text{if } |z_n| \le r, \\ I_q(R, p_2, z_n), & \text{if } |z_n| > r, \end{cases} \tag{35}$$

where $r > 0$,

– switching using $|p(z)|$

$$z_{n+1} = \begin{cases} I_q(R, p_1, z_n), & \text{if } |\overline{p}(z_n)| \le r, \\ I_q(R, p_2, z_n), & \text{if } |\overline{p}(z_n)| > r, \end{cases} \tag{36}$$

where $r > 0$ and $\overline{p}$ is the polynomial with greater degree. Of course we can change the order of p_1 and p_2 in (36).

The last group of switching processes is the switching between convergence tests. Similarly like in the other groups in the polynomiograph's generation algorithm, we fix all the parameters except the convergence test. Depending on the switching method, we choose two or more convergence test $T^0_{t_0}, T^1_{t_1}, \ldots, T^{m-1}_{t_{m-1}}$ ($m \geq 2$). Now, we define a convergence test that consists of switching process in the following way:

– switching modulo m

$$T(z_n, z_{n+1}) = \begin{cases} T^0_{t_0}(z_n, z_{n+1}), & \text{if } n \mod m = 0, \\ T^1_{t_1}(z_n, z_{n+1}), & \text{if } n \mod m = 1, \\ \ldots \\ T^{m-1}_{t_{m-1}}(z_n, z_{n+1}), & \text{if } n \mod m = m-1, \end{cases} \tag{37}$$

– switching using $|z|$

$$T(z_n, z_{n+1}) = \begin{cases} T^0_{t_0}(z_n, z_{n+1}), & \text{if } |z_{n+1}| \leq r, \\ T^1_{t_1}(z_n, z_{n+1}), & \text{if } |z_{n+1}| > r, \end{cases} \tag{38}$$

where $r > 0$,

– switching using $|p(z)|$

$$T(z_n, z_{n+1}) = \begin{cases} T^0_{t_0}(z_n, z_{n+1}), & \text{if } |p(z_{n+1})| \leq r, \\ T^1_{t_1}(z_n, z_{n+1}), & \text{if } |p(z_{n+1})| > r, \end{cases} \tag{39}$$

where $r > 0$.

4 Examples

In this section, we present some examples of polynomiographs obtained with the help of the switching processes introduced in Sect. 3. Moreover, we present generation times of those polynomiographs.

The software for polynomiographs' generation has been implemented in Processing, a programming language based on Java, and it has been run on a computer with the following specification: Intel Core i5-4570 processor, 16 GB RAM, 64-bit version of Windows 10. All the presented polynomiographs have been generated in 600 × 600 pixels resolution. Moreover, each polynomiograph has been generated 30 times to decrease the influence of other processes and the virtual Java machine running on the computer. Then, the worst and the best times have been rejected to obtain the average generation time.

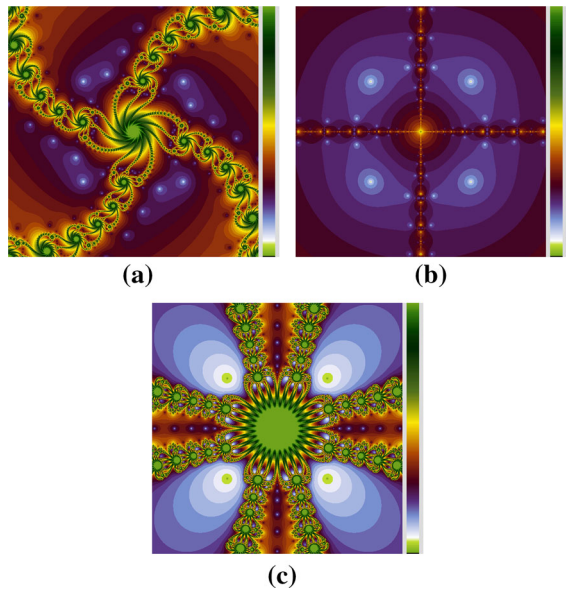

Fig. 2 Polynomiographs obtained with the use of different root finding methods. **a** Relaxed Newton, **b** relaxed Halley **c** E_3

We start with examples of the first group of switching processes, namely switching of the root finding methods. Fig. 2 presents polynomiographs obtained with the relaxed Newton's method with $\lambda = 0.75 + 0.5i$, the relaxed Halley's method with $\lambda = 0.5$ and the E_3 method. The common parameters used to generate these polynomiographs were as follows: $p(z) = z^4 + 4$, $A = [-2.5, 2.5]^2$, $K = 40$, Picard's iteration, convergence test (23) with $\varepsilon = 0.001$. The generation times of these polynomiographs are presented in Table 1. The result of switching the three root finding methods from Fig. 2 using the modulo switching (27) is presented in Fig. 3, and the generation times are given in Table 2. From the images, we see that the polynomiographs obtained with the help of this switching process differ in a significant way from the original polynomiographs. The change in shape is equivalent to the change of the dynamics of the corresponding dynamical system. In each pattern obtained with this switching process, we can find some features of the original patterns, e.g., swirls from the Newton method, the shape of the central part of the E_3 method. Looking at the times from Tables 1 and 2, we see that the generation times for

Table 1 Generation times of the polynomiographs from Fig. 2

Method	Time (s)
Relaxed Newton	0.975
Relaxed Halley	0.905
E_3	2.093

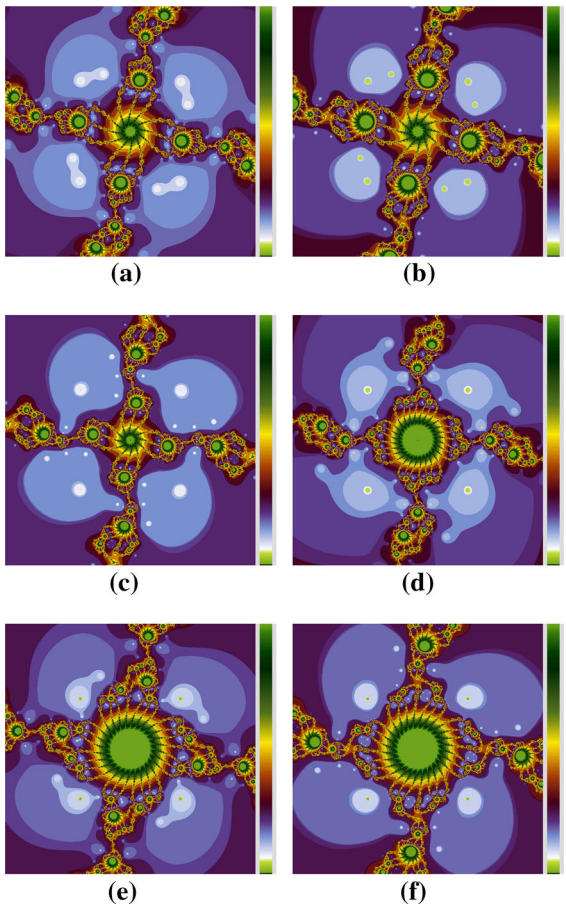

Fig. 3 Polynomiographs obtained with the use of switching of the relaxed Newton method (N), the relaxed Halley method (H) and the E_3 method from Fig. 2 using the modulo switching process (27). **a** N–H–E_3, **b** N–E_3–H, **c** H–N–E_3, **d** H–E_3–N, **e** E_3–N–H, **f** E_3–H–N

polynomiographs obtained using this type of switching are between the times obtained for the original root finding methods used in the switching process.

In the next example, we used switching process of the root finding methods using the $|z|$, i.e., Eq. (28). The original polynomiographs obtained with the help of the relaxed Newton ($\lambda = 0.75$) and the relaxed Halley ($\lambda = 1.0$) methods are presented in Fig. 4. The common parameters for these two polynomiographs were as follows: $p(z) = z^4 + 4$, $A = [-2.5, 2.5]^2$, $K = 40$, Picard's iteration, convergence test (23) with $\varepsilon = 0.001$. Table 3 shows generation times of the polynomiographs. The result of switching the two root finding methods using (28) where R_0 is Halley's method and R_1 is Newton's method is presented in Fig. 5. The values of r were as follows: (a) 0.5, (b) 1.0, (c) 1.5, (d) 2.0. The generation times of the polynomiographs are presented in Table 4. From the polynomiographs, we can observe that for different values of r we obtain patterns that differ from the original ones. Moreover, we see that for low values of r the pattern is very similar to the pattern obtained with the help of R_1 method. When we increase the value of r, then the pattern is more similar to the pattern obtained with the help of R_0 method. Comparing the times from Tables 3 and 4, we see that similar to the previous case the times for polynomiographs obtained with the switching process are between the times obtained for the original root finding methods. Moreover, the greater the value of r the lower the time, which gets closer to the time for the relaxed Halley method.

Table 2 Generation times of the polynomiographs from Fig. 3

Methods	Time (s)
N–H–E_3	1.034
N–E_3–H	1.129
H–N–E_3	1.026
H–E_3–N	1.065
E_3–N–H	1.181
E_3–H–N	1.223

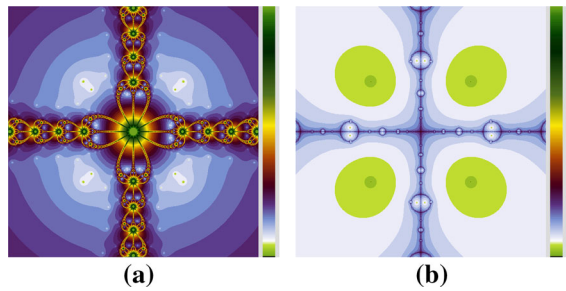

Fig. 4 Polynomiographs obtained with the use of different root finding methods. **a** Relaxed Newton, **b** relaxed Halley

Table 3 Generation times of the polynomiographs from Fig. 4

Method	Time (s)
Relaxed Newton	0.569
Relaxed Halley	0.378

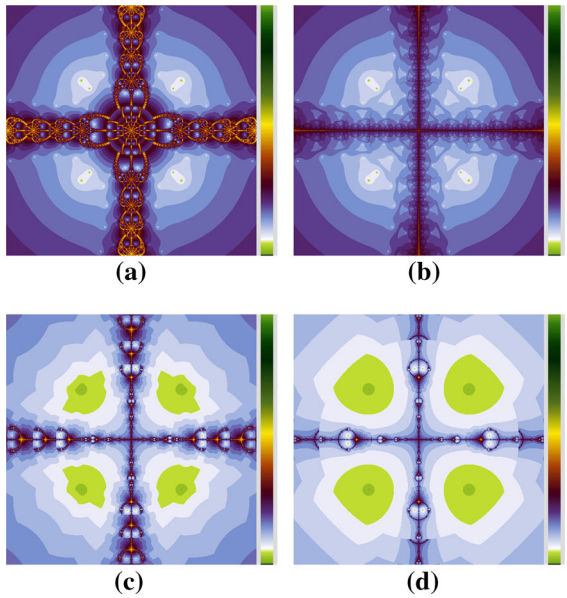

Fig. 5 Polynomiographs obtained with the use of switching of root finding methods using (28) and different values of r. **a** $r = 0.5$, **b** $r = 1.0$, **c** $r = 1.5$, **d** $r = 2.0$

Table 4 Generation times of the polynomiographs from Fig. 5

r	Time (s)
0.5	0.554
1.0	0.510
1.5	0.466
2.0	0.423

In the last example of the first group, we used the switching process of root finding methods given by (29). In the switching process, we used the same root finding methods and their parameters that were used in Fig. 4. The results of switching Newton's method (R_0) and Halley's method (R_1) for different values of r are presented in Fig. 6. The values of r were as follows: (a) 1.0, (b) 3.0, (c) 4.0, (d) 5.0. The generation times of the polynomiographs are presented in Table 5. Similar to the previous example, the shape of polynomiograph changes with the change of r. The lower the value of

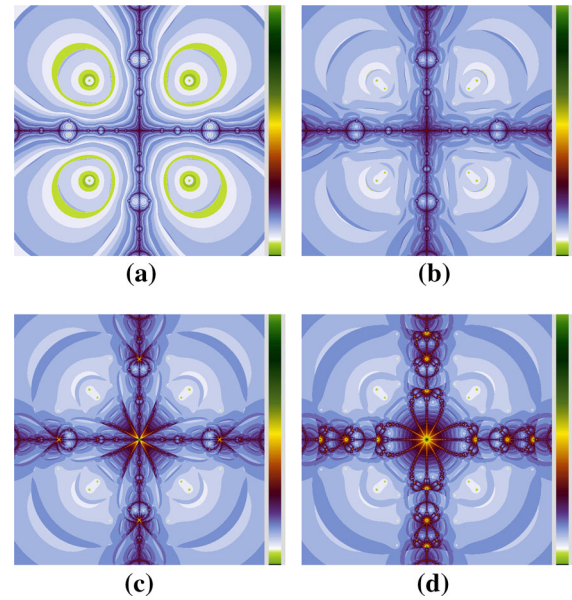

Fig. 6 Polynomiographs obtained with the use of switching of root finding methods using (29) and different values of r. **a** $r = 1.0$, **b** $r = 3.0$, **c** $r = 4.0$, **d** $r = 5.0$

Table 5 Generation times of the polynomiographs from Fig. 6

r	Time (s)
1.0	0.555
3.0	0.612
4.0	0.648
5.0	0.680

r the more the pattern reminds polynomiograph of R_1. The change of the polynomiograph has different character than in the example with the switching process using $|z|$, whereas the generation times starting from $r = 3.0$ are greater than both times obtained for the original root finding methods (Table 3). Moreover, we see that the greater the value of r the greater the time.

The second group of examples presents polynomiographs generated by using the switching processes of iteration methods. In all the examples in this group, we use the same two iterations. The first iteration is Noor's iteration with $\alpha_n = 0.15$, $\beta_n = 0.6$ and $\gamma_n = 0.4$, and the second iteration is CR iteration with $\alpha_n = 0.8$, $\beta_n = 0.8$, $\gamma_n = 0.8$. The other parameters used to generate the polynomiographs from Fig. 7 were as follows: $p(z) = z^3 - 1$, $A = [-2.5, 2.5]^2$, $K = 35$, relaxed Newton's root finding method with

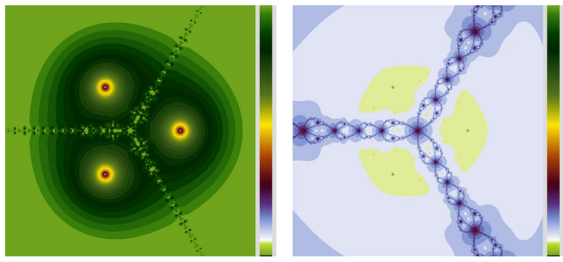

Fig. 7 Polynomiographs obtained with the use of different iteration methods. **a** Noor, **b** CR

Table 6 Generation times of the polynomiographs from Fig. 7

Iteration	Time (s)
Noor	4.935
CR	0.683

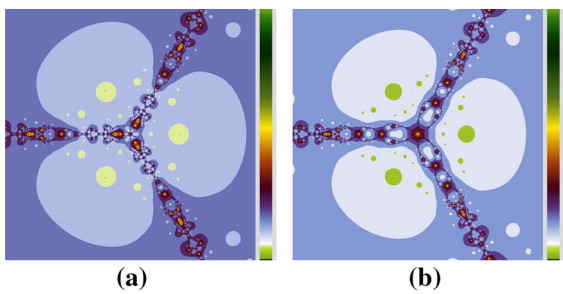

Fig. 8 Polynomiographs obtained with the use of switching of iteration methods using the modulo switching process (30). **a** Noor–CR, **b** CR–Noor

Table 7 Generation times of the polynomiographs from Fig. 8

Iterations	Time (s)
Noor–CR	1.105
CR–Noor	0.977

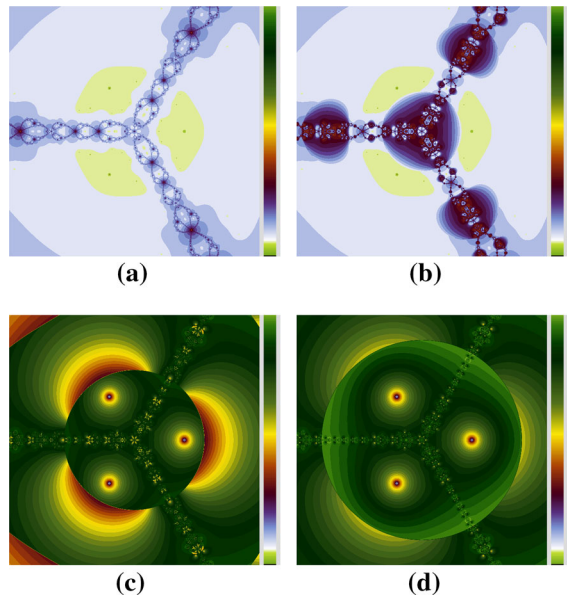

Fig. 9 Polynomiographs obtained with the use of switching of iteration methods using (31) and different values of r. **a** $r = 0.2$, **b** $r = 0.8$, **c** $r = 1.4$, **d** $r = 2.0$

$\lambda = 1$, convergence test (23) with $\varepsilon = 0.001$. The generation times of these polynomiographs are presented in Table 6.

The first switching process of iteration methods is the switching using modulo (30). The results of switching iterations from Fig. 7 using this switching process and their generation times are presented in Fig. 8 and Table 7, respectively. In Fig. 8a we see switching of Noor iteration (in even steps) and CR iteration (in odd steps), and in Fig. 8b switching in the reversed order, namely CR iteration (in even steps) and Noor iteration (in odd steps). From the images, we see that in both cases the shape of polynomiograph has changed in a significant way forming new patterns. And from the tables with the times, we see that the generation times are much lower than the time for the CR iteration and are greater than for the Noor iteration.

In the next example, we used switching process given by (31). As the iteration $I^0_{q_0}$, we took the Noor iteration and as the iteration $I^1_{q_1}$ we took the CR iteration, both with the parameters used to generate polynomiographs in Fig. 7. The values of r used to generate the images were as follows: (a) 0.2, (b) 0.8, (c) 1.4, (d) 2.0. The generation times are given in Table 8. The polynomiographs change their shape from shape very similar to the one obtained with the $I^1_{q_1}$ iteration (for low values of r) to a shape remaining the one obtained with the $I^0_{q_0}$ iteration (for high values of r). The generation times are between the times obtained for the original iterations that were used in the switching process. Moreover, the greater the value of r the greater the time. This was to be expected looking at the polynomiographs, because the greater the value of r the more closely the pattern resembles the pattern obtained for the Noor iteration for which the time was high.

Table 8 Generation times of the polynomiographs from Fig. 9

r	Time (s)
0.2	0.736
0.8	0.867
1.4	4.301
2.0	4.832

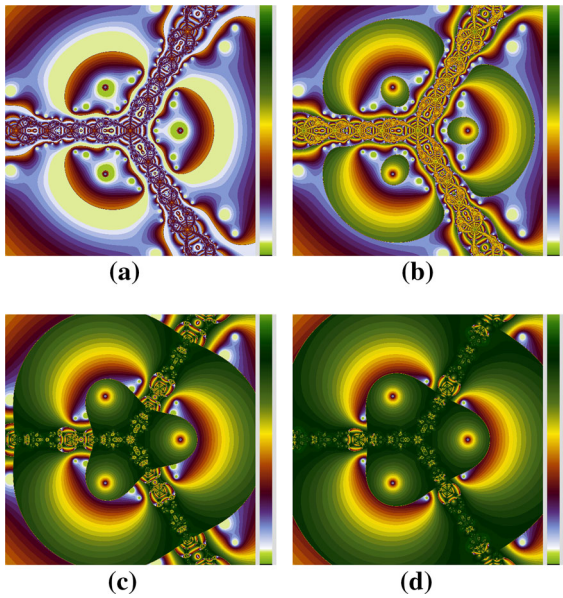

Fig. 10 Polynomiographs obtained with the use of switching of iteration methods using (32) and different values of r. **a** $r = 0.2$, **b** $r = 0.8$, **c** $r = 1.4$, **d** $r = 2.0$

The second group of examples ends with an example of polynomiographs obtained with the switching process (32). Similarly like in the second example of this group, we used Noor's iteration as $I^0_{q_0}$ and CR iteration as $I^1_{q_1}$. The obtained polynomiographs are presented in Fig. 10, and the values of r used to generate them were as follows: (a) 0.2, (b) 0.8, (c) 1.4, (d) 2.0. The generation times of these polynomiographs are presented in Table 9. From the obtained images, we clearly see that using different values of r we obtain various shapes of polynomiographs. The shapes are different from the original ones and from the ones obtained with the switching process that used $|z|$. The generation times in this case also are between the times obtained for the original iterations.

The third group of examples consists of polynomiographs obtained with the switching of polynomials. As in the previous groups, we start with the switching using modulo—equation (34). Figure 11 presents polynomiographs obtained for two different polynomials: (a) $p_1(z) = z^8 + 3z^4 - 4$, (b) $p_2(z) = z^4 + 4$, and Table 10 presents the times of their generation. The sets of roots for both the polynomials were as follows: $S_1 = \{-1, -1-i, -1+i, -i, i, 1, 1-i, 1+i\}$, $S_2 = \{-1-i, -1+i, 1-i, 1+i\}$. We clearly see that $S_2 \subsetneq S_1$. The other common parameters used to generate the polynomiographs were as follows: $A = [-4, 4]^2$, $K = 35$, Picard's iteration, relaxed Newton's method with $\lambda = 1$, convergence test (23) with $\varepsilon = 0.001$. Figure 12 presents the result of switching polynomials p_1 and p_2 using (34) and Table 11 times of its generation. In Fig. 12a we see polynomiograph obtained using p_1 in the even steps and p_2 in the odd steps of the switching process, and in Fig. 12b the order of the polynomials was reversed, i.e., p_2 was used in the even steps and p_1 in the odd steps. The overall shape of both the polynomiographs reminds the shape of polynomiograph obtained for the polynomial with the lower degree, i.e., p_2. However, the polynomiographs obtained with the switching reveal much more details, what makes them more interesting. The generation times are between the times obtained for the original polynomials, and they are closer to the time for p_1.

Table 9 Generation times of the polynomiographs from Fig. 10

r	Time (s)
0.2	1.465
0.8	2.811
1.4	3.930
2.0	4.674

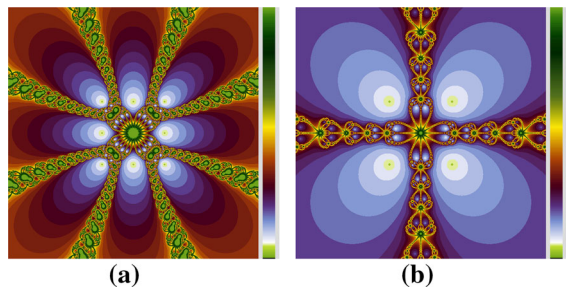

Fig. 11 Polynomiographs obtained with the use of different polynomials. **a** $p_1(z) = z^8 + 3z^4 - 4$, **b** $p_2(z) = z^4 + 4$

Table 10 Generation times of the polynomiographs from Fig. 11

Polynomial	Time (s)
p_1	1.301
p_2	0.470

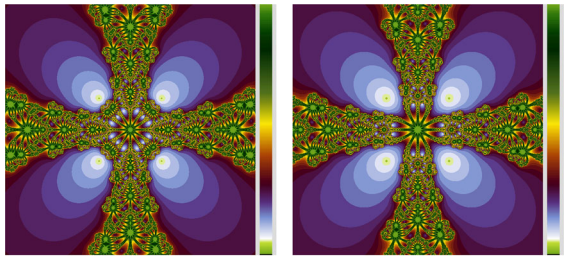

Fig. 12 Polynomiographs obtained with the use of switching of polynomials using the modulo switching process (34). **a** p_1–p_2, **b** p_2–p_1

Table 11 Generation times of the polynomiographs from Fig. 12

Polynomials	Time (s)
p_1–p_2	1.117
p_2–p_1	1.042

For the next example, we used switching process of the polynomials given by (35). The original polynomiographs obtained for $p_1(z) = z^4 - 1$ and $p_2(z) = z^2 - 1$ are presented in Fig. 13, and in Table 12 their generation times are given. The sets of roots for these polynomials are the following: $S_1 = \{-1, -i, 1, i\}$, $S_2 = \{-1, 1\}$. We see that $S_2 \subsetneq S_1$. The other common parameters used to generate the polynomiographs were as follows: $A = [-4, 4]^2$, $K = 35$, Picard's iteration, relaxed Newton's method with $\lambda = 1$, convergence test (23) with $\varepsilon = 0.001$. Figure 14 presents polynomiographs obtained with the switching of p_1 and p_2 for the following values of r: (a) 0.5, (b) 1.2999999523, (c) 2.0999999046, (d) 2.5. The generation times of the polynomiographs are presented in Table 13. The pattern from Fig. 14a is similar to the original pattern from Fig. 13b, and the only difference is in the central part of the polynomiograph. When we increase the value of r, we observe that the pattern changes and is more similar to the original pattern from Fig. 13a. Comparing the times from Tables 12 and 13, we see that the generation times of the polynomiographs obtained with the

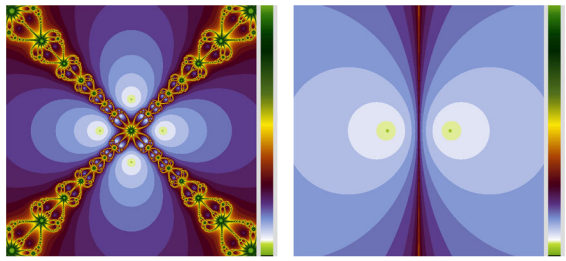

Fig. 13 Polynomiographs obtained with the use of different polynomials. **a** $p_1(z) = z^4 - 1$, **b** $p_2(z) = z^2 - 1$

Table 12 Generation times of the polynomiographs from Fig. 13

Polynomial	Time (s)
p_1	0.588
p_2	0.224

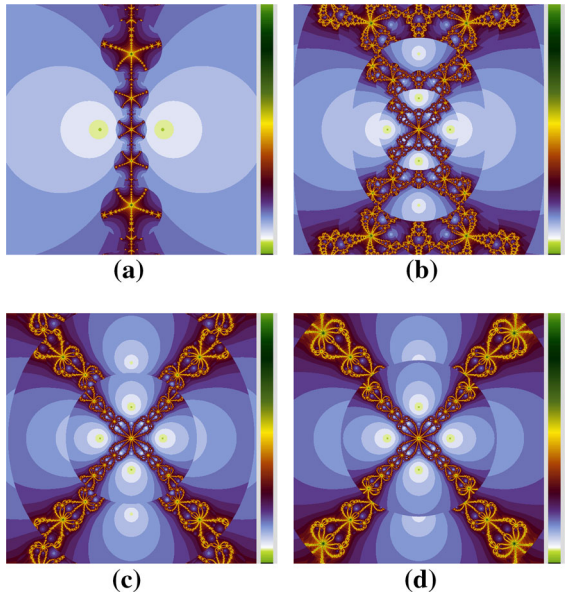

Fig. 14 Polynomiographs obtained with the use of switching of polynomials using (35) and different values of r. **a** $r = 0.5$, **b** $r = 1.2999999523$, **c** $r = 2.0999999046$, **d** $r = 2.5$

switching process are between the times obtained for the original polynomials and that the greater the value of r the greater the time.

In the last example of the third group, we used switching process of polynomials given by (36). We used the same polynomials and common parameters as in Fig. 13, but this time we changed the order of the

Table 13 Generation times of the polynomiographs from Fig. 14

r	Time (s)
0.5	0.262
1.2999999523	0.475
2.0999999046	0.492
2.5	0.529

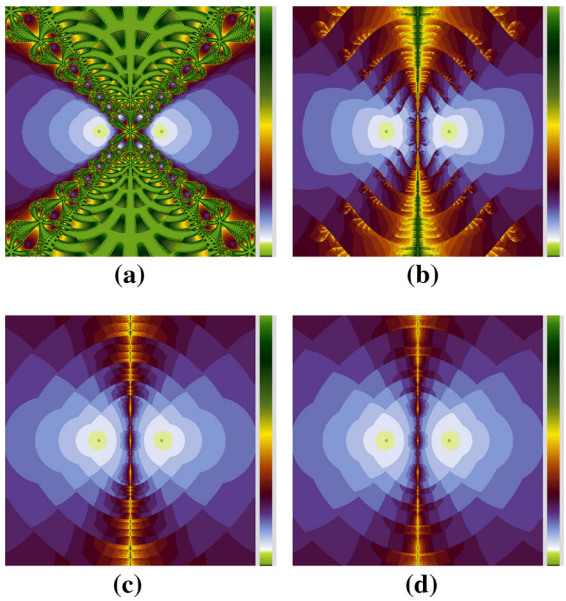

Fig. 15 Polynomiographs obtained with the use of switching of polynomials using (36) and different values of r. **a** $r = 0.8000000119$, **b** $r = 1.5$, **c** $r = 3.5999999046$, **d** $r = 5.0$

polynomials, i.e., $p_1(z) = z^2 - 1$ and $p_2(z) = z^4 - 1$. Figure 15 presents polynomiographs obtained with the following values of the r parameter: (a) 0.8000000119, (b) 1.5, (c) 3.5999999046, (d) 5.0, and Table 14 presents times of their generation. From the obtained images, we see that the polynomiographs differ from the original ones in a significant way. We can observe that the most noticeable change of the shape is in the central part of the polynomiograph. Moreover, we can notice that for high values of r the change of its value does not affect the shape in a significant way like in the case of the low values. Looking at the tables with the times (Tables 12 and 14), we see that the times obtained for polynomiographs with switching are greater than the times obtained for the original polynomials. Moreover, we see that the greater the value of r the lower the time.

Table 14 Generation times of the polynomiographs from Fig. 15

r	Time (s)
0.8000000119	1.755
1.5	0.796
3.5999999046	0.634
5.0	0.598

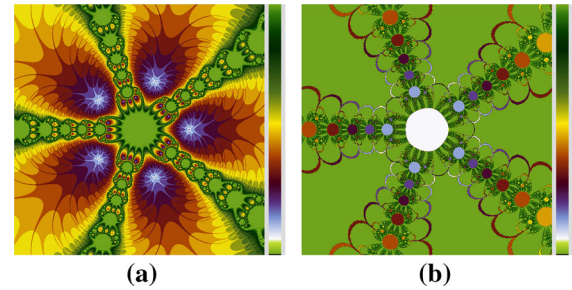

Fig. 16 Polynomiographs obtained with the use of different convergence tests. **a** Test (40), **b** test (41)

The last group of examples presents polynomiographs obtained using the switching of convergence tests. We start with switching using (37). For this type of switching, we used the following convergence tests:

$$||z_{n+1}|^2 - |z_n|^2| < 0.001, \tag{40}$$

$$\left| \frac{0.05}{|z_{n+1}|^2} - \frac{0.045}{|z_n|^2} \right| < 0.001. \tag{41}$$

Polynomiographs obtained for these two convergence tests and the following common parameters: $p(z) = z^5 - 1$, $A = [-2.5, 2.5]^2$, $K = 15$, Picard's iteration, the relaxed Newton's method with $\lambda = 1$ are presented in Fig. 16, whereas their generation times are presented in Table 15. The result of switching between (40) and (41) and between (41) and (40) using (37) are presented in Fig. 17a and b, respectively. Both polynomiographs are different from the original ones. Moreover, they reveal features of both of the original patterns, but are more similar to the pattern from Fig. 16a. When we look at the generation times, we see that the times for polynomiographs obtained with the considered switching process are between the times obtained for the original convergence tests (Table 16).

In the example with switching using (38), for the generation of polynomiographs, we used the same parameters as in the example for (37), but with other convergence tests:

Table 15 Generation times of the polynomiographs from Fig. 16

Test	Time (s)
(40)	0.586
(41)	0.893

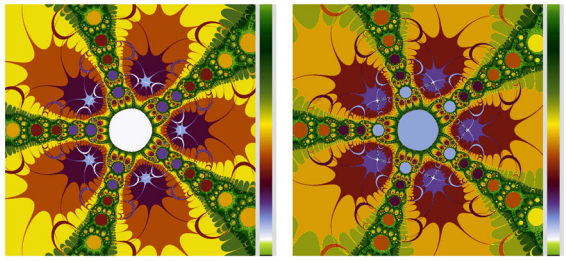

Fig. 17 Polynomiographs obtained with the use of switching of convergence tests using the modulo switching process (37). **a** (40)–(41), **b** (41)–(40)

Table 16 Generation times of the polynomiographs from Fig. 17

Tests	Time (s)
(40)–(41)	0.608
(41)–(40)	0.614

$$|0.4\Re(z_{n+1} - z_n)|^2 < 0.001 \\ \wedge |\Im(z_{n+1} - z_n)|^2 < 0.001, \quad (42)$$

$$|z_{n+1} - z_n| < 0.001, \quad (43)$$

where $\Re(z)$, $\Im(z)$ are the real and the imaginary part of z, respectively. The polynomiographs and the generation times obtained for these two convergence tests are presented in Fig. 18 and Table 17, respectively, whereas Fig. 19 presents polynomiographs obtained with the switching process (38), where test (42) is T^0 and test (43) is T^1. The values of r were as follows: (a) 0.999, (b) 0.9999, (c) 1.0, (d) 1.0001. The generation times of the polynomiographs are given in Table 18. From the polynomiographs, we can observe that the shape changes in the areas where the root finding method converges to some root. Moreover, we see that this type of switching is very sensitive to the change of r, i.e., very small change of the value causes that the shape changes in a significant way. The generation times of polynomiographs obtained using the switching process are greater than the times obtained for the original con-

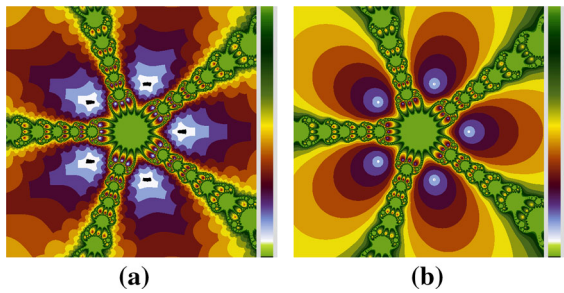

Fig. 18 Polynomiographs obtained with the use of different convergence tests. **a** Test (42), **b** test (43)

Table 17 Generation times of the polynomiographs from Fig. 18

Test	Time (s)
(42)	0.491
(43)	0.586

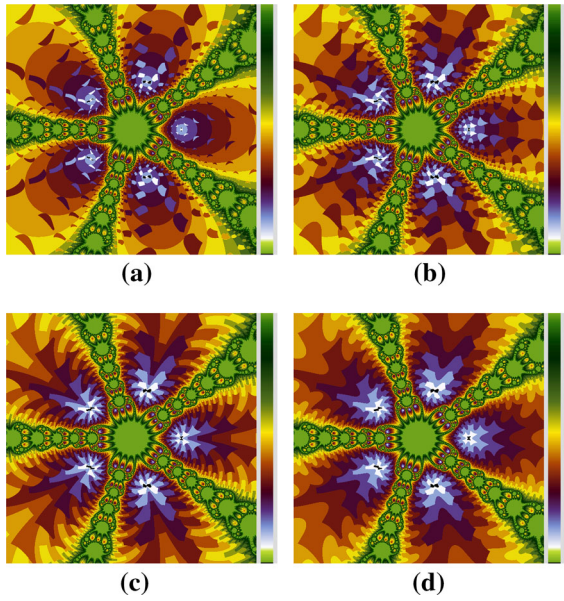

Fig. 19 Polynomiographs obtained with the use of switching of convergence tests using (38) and different values of r. **a** $r = 0.999$, **b** $r = 0.9999$, **c** $r = 1.0$ **d** $r = 1.0001$

vergence tests except the time for $r = 1.0001$ which is lower than the time for (43).

The last example presents polynomiographs obtained with the switching process given by (39). Parameters used in the example were the same like in the previous two examples except the convergence tests used in the

Table 18 Generation times of the polynomiographs from Fig. 19

r	Time (s)
0.999	0.629
0.9999	0.618
1.0	0.598
1.0001	0.584

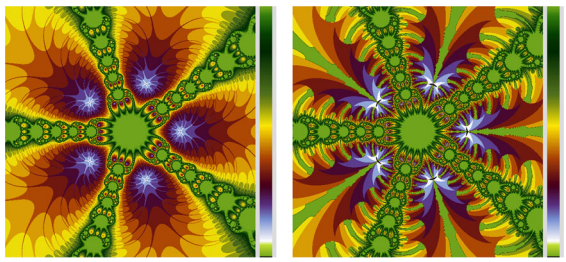

Fig. 20 Polynomiographs obtained with the use of different convergence tests. **a** Test (44), **b** test (45)

switching process. The tests were as follows:

$$|0.01(z_{n+1} - z_n)| + |0.029|z_{n+1}|^2 - 0.03|z_n|^2| < 0.001, \quad (44)$$

$$||z_{n+1}|^2 - |z_n|^2| < 0.001. \quad (45)$$

Fig. 20 presents polynomiographs obtained with these two convergence tests and Table 19 theirs generation times. The result of using switching process given by (39), where test (44) is T^0 and test (45) is T^1, is presented in Fig. 21. The values of r used to generate these polynomiographs were as follows: (a) 0.000000001, (b) 0.0001, (c) 0.0099999998, (d) 0.1000000015. The generation times of the polynomiographs are given in Table 20. Similar to the previous example, we see that the shape of polynomiograph changes in the areas where the root finding method converges to a root. Also we can observe that this type of switching is very sensitive to the change of r. From Tables 19, 20 we see that the generation times of polynomiographs obtained with the use of switching are greater than the times obtained for the original convergence tests used in the switching process.

Table 19 Generation times of the polynomiographs from Fig. 20

Test	Time (s)
(44)	0.671
(45)	0.586

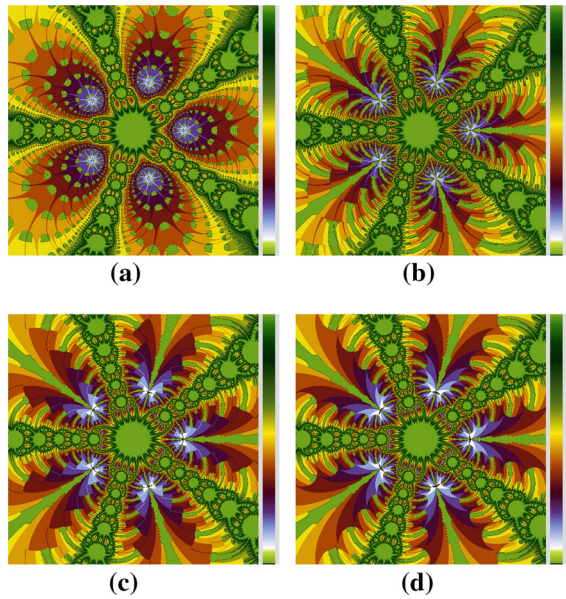

Fig. 21 Polynomiographs obtained with the use of switching of convergence tests using (39) and different values of r. **a** $r = 0.000000001$, **b** $r = 0.0001$, **c** $r = 0.0099999998$, **d** $r = 0.1000000015$

Table 20 Generation times of the polynomiographs from Fig. 21

r	Time (s)
0.000000001	1.026
0.0001	1.168
0.0099999998	1.018
0.1000000015	1.021

5 Conclusions

In this paper, we presented some modifications of the polynomiograph's generation algorithm. The modifications rely on the use of different switching processes, i.e., switching of root finding methods, iterations, polynomials and convergence tests. The obtained polynomiographs look quite different in comparison with those obtained with the help of the standard polynomio-

graph's generation algorithm. The results of this paper can further extend the possibilities of the existing polynomiographs' generation software and therefore can be used to create paintings, in carpet or tapestry design [11].

Recently, García-Morales introduced the notion of digit function [6]. This function is a building block for other two notions. In [6] García-Morales has shown that using the digit function we can decompose any real or complex valued function (single or multivariable) into a finite set of fractal discontinuous functions. This type of decomposition is called the $p\lambda n$ decomposition. In [5] we find a definition of a generalized bitwise operator. To define this type of operators, the author uses the digit function. Both these notions, $p\lambda n$ decomposition and generalized bitwise operators, probably might extend the possibilities of generating new and diverse polynomiographs. For instance, we could use the $p\lambda n$ decomposition to decompose, e.g., polynomials or convergence tests, that are complex valued functions, and use switching of the parts of these decompositions.

Acknowledgements The author would like to thank the anonymous referees for their careful reading of the manuscript and their fruitful comments and suggestions that helped to improve the paper.

References

1. Ardelean, G.: A comparison between iterative methods by using the basins of attraction. Appl. Math. Comput. **218**(1), 88–95 (2011). doi:10.1016/j.amc.2011.05.055
2. Ashlock, D., Jamieson, B.: Evolutionary exploration of complex fractals. In: Hingston, P., Barone, L., Michalewicz, Z. (eds.) Design by Evolution, Natural Computing Series, pp. 121–143. Springer, Berlin (2008). doi:10.1007/978-3-540-74111-4_8
3. Danca, M.F., Bourke, P., Romera, M.: Graphical exploration of the connectivity sets of alternated Julia sets. Nonlinear Dyn. **73**(1), 1155–1163 (2013). doi:10.1007/s11071-013-0859-y
4. Danca, M.F., Romera, M., Pastor, G.: Alternated Julia sets and connectivity properties. Int. J. Bifurc. Chaos **19**(6), 2123–2129 (2009). doi:10.1142/S0218127409023962
5. García-Morales, V.: Fractal surfaces from simple arithmetic operations. Phys. A: Stat. Mech. Appl. **447**, 535–544 (2016). doi:10.1016/j.physa.2015.12.028
6. García-Morales, V.: The $p\lambda n$ fractal decomposition: nontrivial partitions of conserved physical quantities. Chaos Solitons Fractals **83**, 27–37 (2016). doi:10.1016/j.chaos.2015.11.028
7. Gdawiec, K.: Polynomiography and various convergence tests. In: Skala, V. (ed.) WSCG 2013 Communication Papers Proceedings, pp. 15–20. Plzen, Czech Republic, Vaclav Skala - Union Agency (2013)
8. Gdawiec, K.: Mandelbrot- and Julia-like rendering of polynomiographs. In: Chmielewski, L., Kozera, R., Shin, B.S., Wojciechowski, K. (eds.) Computer Vision and Graphics, Lecture Notes in Computer Science, vol. 8671, pp. 25–32. Springer International Publishing (2014). doi:10.1007/978-3-319-11331-9_4
9. Gdawiec, K., Kotarski, W., Lisowska, A.: Polynomiography based on the non-standard Newton-like root finding methods. Abstr. Appl. Anal. **2015**(797), 594 (2015). doi:10.1155/2015/797594
10. Gilbert, W.: Generalizations of Newton's method. Fractals **9**(3), 251–262 (2001). doi:10.1142/S0218348X01000737
11. Kalantari, B.: Polynomiography and applications in art, education and science. Comput. Graph. **28**(3), 417–430 (2004). doi:10.1016/j.cag.2004.03.009
12. Kalantari, B.: Polynomial Root-Finding and Polynomiography. World Scientific, Singapore (2009). doi:10.1142/9789812811837
13. Lakhtakia, A.: Julia sets of switched processes. Comput. Graph. **15**(4), 597–599 (1991). doi:10.1016/0097-8493(91)90061-L
14. Negi, A., Rani, M., Mahanti, P.: Computer simulation of the behaviour of Julia sets using switching processes. Chaos Solitons Fractals **37**(4), 1187–1192 (2008). doi:10.1016/j.chaos.2006.10.061
15. Pickover, C.: Halley maps for a trigonometric and rational function. Comput. Math. Appl. **17**(1–3), 125–132 (1989). doi:10.1016/0898-1221(89)90153-3
16. Shirriff, K.: Fractals from simple polynomial composite functions. Comput. Graph. **17**(6), 701–703 (1993). doi:10.1016/0097-8493(93)90122-P
17. Varona, J.: Graphics and numerical comparison between iterative methods. Math. Intel. **24**(1), 37–46 (2002). doi:10.1007/BF03025310
18. Wang, D., Liu, S.: On the boundedness and symmetry properties of the fractal sets generated from alternated complex map. Symmetry **8**(2), 7 (2016). doi:10.3390/sym8020007
19. Xing-Yuan, W.: Switched processes generalized Mandelbrot sets for complex index number. Appl. Math. Mech. **24**(1), 73–81 (2003). doi:10.1007/BF02439380
20. Xingyuan, W.: The generalized M sets from a class complex mapping system. Appl. Math. Comput. **175**(2), 1484–1494 (2006). doi:10.1016/j.amc.2005.08.028
21. Yadav, A., Rani, M.: Alternate superior Julia sets. Chaos Solitons Fractals **73**, 1–9 (2015). doi:10.1016/j.chaos.2014.12.008